SEP Central, LET, and SEPT Flight Software

Presenter: Andrew Davis
ad@srl.caltech.edu

11-14-2002

Personnel and Responsibilities

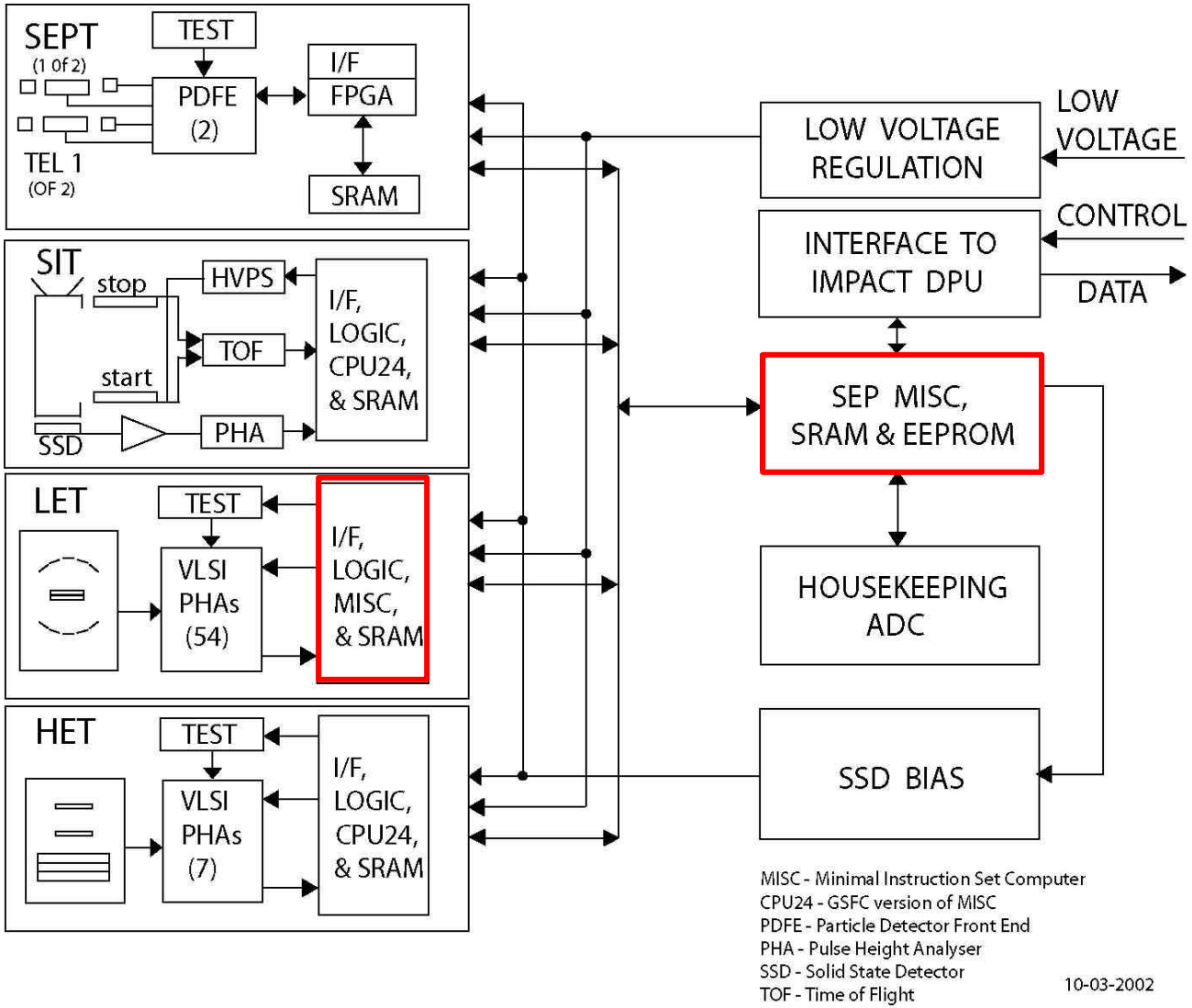
- Caltech – SEP Central, LET, and SEPT flight software
  - Rick Cook, Andrew Davis, Alan Labrador
- JPL – GSE Software
  - Bob Radocinski

Applicable Documents

- **IMPACT Performance Specification**
- **LET/SEP-Central and SEP GSE Software Development Plans** (STEREO-CIT-001.H, STEREO-CIT-012.A)
- **LET/SEP-Central Software Requirements** (STEREO-CIT-002.E)
- **SEPT Operation Control and Data Processing Requirements**
- **IMPACT Intra-Instrument Interface ICD**
- **SEP Sensor Suite ICDs** (four documents, STEREO-CIT-008.A, 009.A, 010.A, 011.A)
- **P24 MISC Processor Manual** (STEREO-CIT-005.A)
- **LET Science Data Frame Format Specification** (STEREO-CIT-003.A)
- **LET Functional Test Plan** (STEREO-CIT-006.A)
- **SEP Commanding and Users Manual** (STEREO-CIT-007.A)

Available from http://sprg.ssl.berkeley.edu/impact/dwc/

And Caltech ftp site: mussel.srl.caltech.edu, in pub/stereo/docs


SEPT
(1 of 2)
TEST
PDFE
(2)
I/F
FPGA
SRAM
TEL 1
(OF 2)
SIT
stop
start
SSD
HVPS
TOF
PHA
I/F,
LOGIC,
CPU24,
& SRAM
LET
TEST
VLSI
PHAs
(54)
I/F,
LOGIC,
MISC,
& SRAM
HET
TEST
VLSI
PHAs
(7)
I/F,
LOGIC,
CPU24,
& SRAM
LOW VOLTAGE
REGULATION
LOW
VOLTAGE
INTERFACE TO
IMPACT DPU
CONTROL
DATA
SEP MISC,
SRAM & EEPROM
HOUSEKEEPING
ADC
SSD BIAS
MISC - Minimal Instruction Set Computer
CPU24 - GSFC version of MISC
PDFE - Particle Detector Front End
PHA - Pulse Height Analyser
SSD - Solid State Detector
TOF - Time of Flight
10-03-2002

LET/SEP-Central Software Development Plan (SDP)

- **LET/SEP-Central Software Development Plan (SDP) approved January 2002 by Caltech and submitted to Project for review.**
- **Project comments received by Caltech October 16 2002.**
  - **VDD's must accompany all software releases.**
  - **Include Forth OS and low-level I/O in SDP and acceptance test plan.**
  - **TRR shall precede formal acceptance test.**
  - **SAR following successful completion of acceptance test.**
  - **Create and maintain a Software Reqs. verification Matrix.**
- **New version of SDP incorporating Project comments was submitted to Project November 8 2002; we expect final approval of SDP shortly…**

Software Design Methodology – Details in SDP

Top-Down and Bottom-Up Approaches running in parallel

- **Top-Down**
  1. Requirements definition and analysis ® Software Requirements Document, SEP sensor suite ICDs, IDPU ICD.
  2. Design Phase ® subsystem definition, design diagrams, algorithms…
  3. Implementation phase
  4. System testing and acceptance testing

- **Bottom-Up**
  1. Develop small test routines – gain familiarity with MISC processor development environment
  2. Gather together a suitable set of tools for MISC software development, including MISC simulator, serial communication software, RCS version control software, etc.
  3. Verify Forth system on MISC with standard Forth software test suite
  4. Prototype onboard processing algorithms to verify feasibility of MISC hardware approach

Software Quality Assurance Measures

- Only the lead engineer is authorized to load flight software into flight MISCs
- Lead engineer maintains flight source code. Developers deliver software modules to lead engineer for incorporation into the flight code
- Checksums calculated during EEPROM burn-in, checked during boot process
- Version numbering and documentation of changes for all source files
- Archiving and backup of all source code each day the code is worked on
- Formal version control during latter stages of implementation phase
- Software walkthroughs at software peer reviews, reports at PDR and CDR
- System and acceptance tests : end-to-end instrument, electronics, and software functional tests done using accelerator tests, radiation sources, built-in self-test routines and functional test procedures (see LET Functional Test Plan), and simulation data.
- After the beginning of integration and test with flight hardware, Version Description Documents (VDDs) shall accompany all Software releases.

Software Acceptance Test Plan – Details in SDP

- Formal acceptance tests will be performed on the LET sensor, including the flight software, during the SEP integration and test phase. Similarly, SEP-Central and its associated software will undergo acceptance tests as a unit, and as part of the SEP sensor suite.
- A Test Readiness Review shall precede formal acceptance testing.
- The software acceptance tests shall be designed to verify each of the software functional requirements as called out in the requirements documents, and also the functions provided by the Forth operating system, the I/O API, and the multitasking software running on the MISC.
- Software requirements verification matrices shall be created and maintained by to aid in verifying that the software meets the requirements. Preliminary version are included later in this presentation.
- During environmental tests and suite level testing, more experience and test time with the flight software and real sensor data will be gained, and changes are expected. Any subsequent changes in the flight software will result in a repeat of these acceptance tests.
- The LET Functional Test Plan document describes tests that will be used during software acceptance testing.
- A Software Acceptance Review shall follow the successful completion of the acceptance tests.

SEP Central and SEPT Flight Software

Presenter: Andrew Davis
ad@srl.caltech.edu

SEP Central Flight Software Requirements and Verification Matrix (1)

Description	Requirements	Verification
Power-on/Reset	Boot SEP Central from EEPROM	Verify successful boot of SEP central and SEP sensors after power-on and reset. Verify EEPROM checksums.
	Boot LET, HET, and SIT	
RAM and EEPROM Patches	Implement RAM and EEPROM patches uploaded by command from GSE	Upload EEPROM patch, write EEPROM, verify checksum. Similar for RAM.
Command Interface	Receive command messages from IDPU (<= 1 per 7 msec)	Send series of commands, verify receipt of appropriate command responses and telemetry.
	Route LET, HET, and SIT commands to those sensors.	Send commands to each sensor from GSE, verify that correct command responses are received. Check housekeeping/status data to verify that command executed properly.

SEP Central Flight Software Requirements and Verification Matrix (2)

Description	Requirements	Verification
Command Interface (continued)	Buffer command responses and send them to IDPU in cmd response packets	Send series of commands targeted to SEP Central and to sensors; verify receipt of appropriate command responses and telemetry.
	Execute SEP-Central and SEPT commands.	
Data Acquisition	Receive science, beacon, and housekeeping CCSDS packets from LET, HET, SIT.	Verify that packets from LET, HET, and SIT are present in telemetry from SEP Central.
	Acquire science data from SEPT-E and SEPT-NS.	Verify that packets from SEPT are present in telemetry from SEP Central.
	Acquire common SEP housekeeping data from SEP HK ADC.	Verify presence of housekeeping data in HK messages telemetered by SEP Central.

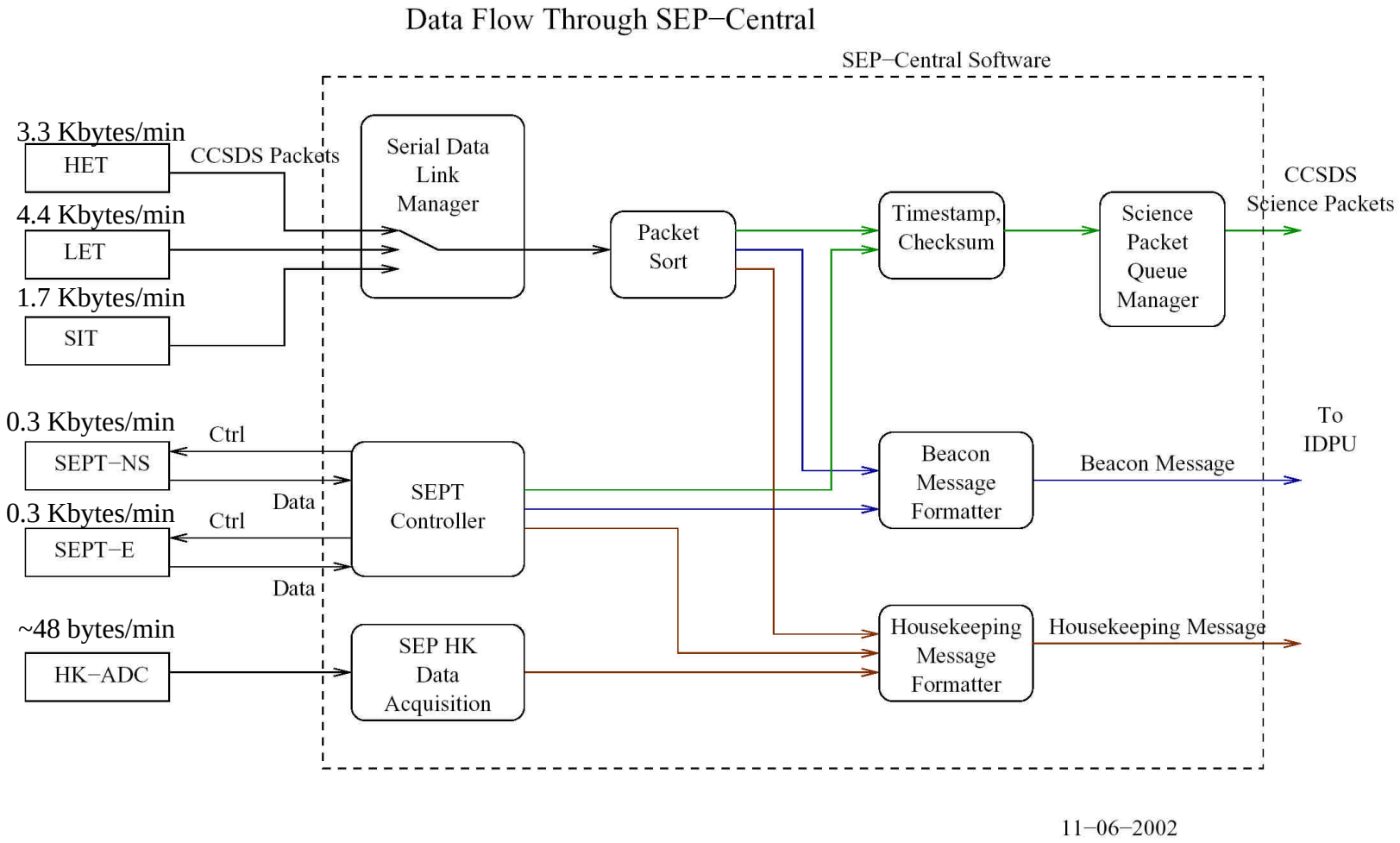
SEP Central Flight Software Requirements and Verification Matrix (3)

Description	Requirements	Verification
SEPT Operation and Data Processing	Manage operation of SEPT-NS and SEPT-E units.	Send commands to cycle SEPT operational modes. Verify science and housekeeping telemetry.
	Log-compress and format SEPT science data and extract SEPT beacon data.	Verify SEPT science and beacon telemetry.
Data Formatting	Add Timestamps and compute checksums for science and cmd response packets.	Verify timestamps and checksums in packets telemetered by SEP Central.
	Build SEP housekeeping and beacon data messages.	Verify contents and format of housekeeping and beacon telemetry.

SEP Central Flight Software Requirements and Verification Matrix (4)

Description	Requirements	Verification
Transmit Data to IDPU	Each minute, transmit science, beacon, and housekeeping packets/messages to IDPU, on a predefined schedule.	Verify that timing of packets/messages telemetered by SEP Central is correct.
Timing	Receive UT time and sample clock messages from IDPU.	Verify that timestamps on SEP telemetry packets are correct, and that packets received from LET, HET, and SIT are properly time-synced.
	Provide 1-second frame sync and minute-marker signals to LET, HET, SIT.	

SEP Central Flight Software Functional Flow Diagram



SEP Central Command Handling

- SEP commands originate on the ground as variable-length CCSDS telecommand packets
- IMPACT IDPU passes SEP command packets to SEP Central after stripping off the CCSDS header
- All necessary routing information is contained in the body of SEP command packets
- Discrete commands: executed by hardwired logic
- Data commands for LET, HET and SIT: SEP Central manages the multiplexed serial command link to LET,HET and SIT based on instructions contained in the command packets.
- Data commands for SEPT and SEP Central are executed directly by SEP Central

Process Flow for a LET Command (same for HET and SIT)

- SEP Central receives a LET-CMD instruction, indicating that a command for LET follows
- SEP Central initializes serial command link with LET
- SEP Central routes subsequent command characters to LET
- LET echoes command characters as part of command response
- SEP Central detects LET command termination character, and terminates LET commanding mode. All subsequent command characters are interpreted by SEP Central, until another LET-CMD instruction (or HET-CMD, or SIT-CMD) is received
- All command responses are buffered by SEP Central and transmitted to IDPU in SEP command response CCSDS packets.

SEPT Flight Data Processing by SEP Central

- The procedures for controlling and acquiring data from SEPT-NS and SEPT-E units are described in the *SEPT Operation Control and Data Processing Requirements* document
- Once per minute, SEP Central will send a series of commands to the SEPT units, reading out science, housekeeping and status data and initiating a new data acquisition interval
- SEP Central will logarithmically compress 32 SEPT 24-bit rates to 12 bits
- Certain counters (energy bins) will be summed for inclusion in SEPT beacon data

SEP Central Interrupt Handling

Interrupt	Max. Time to Service	Max. Service Execution Time	Synchronicity
cmd rec'd from IDPU	7 ms	10 us	ASYNC
Data sent to IDPU	34 us	10 us	SYNC
HET, LET, SIT cmd sent	3 ms	10 us	ASYNC
HET, LET, SIT reply rec'd	170 us	10 us	ASYNC
SEPT cmd sent	~ 1 ms	10 us	SYNC
SEPT data rec'd	170 us	10 us	ASYNC
Timer Done	1 ms	~ 100 us	SYNC

SEP Central CPU and RAM Resource Margins

Task	Processor Cycles, %	Code Size, kwords*	Buffer Memory, kwords*
Operating System	<1%	2	2
Data Acquisition	<1%	1	10
SEPT Data Processing	<1%	2	4
Data Formatting	<1%	4	1
Beacon Data Processing	<1%	2	1
Command Processing	<1%	2	1
Total	**<6%**	**13**	**19**
Available	100%	128kwords	
% of Capacity	**<6%**	**25%**	
Margin	**>1567%**	**300%**	

*Note: for the MISC, one word = 3 bytes

Note: the main reason to run SEP Central MISC @ 4 MHz is so that interrupt service routines complete quickly

SEP Central EEPROM Resource Margins

- SEP Central is equipped with 256kwords of EEPROM (1 word=24 bits)
- The large matrix lookup tables used by LET, HET and SIT can be compressed by a factor of three before they are written to EEPROM, so EEPROM requirements are not as large as RAM requirements.

Sensor	Data type	Requirement (kwords)
HET	Code	8
	Tables	8
LET	Code	13
	Tables	16
SIT	Code	8
	Tables	8.3
SEP Central	Code	13
	Tables	8
Total		82.3 kwords = 32% of capacity ® Margin = 212%

SEP Central Flight Software Status

- ✓ **Definition of requirements, including SEPT data processing requirements (*SEP Central Software Reqs doc, SEPT Operation Control and Data Processing requirements*)**
- ✓ **Definition of interface with IMPACT IDPU (*IMPACT intra-instrument ICD*)**
- ✓ **Definition of interface with LET, HET, SIT and SEPT (*SEP sensor-suite ICDs*)**
- ✓ **Forth OS and multitasking environment written and tested**